Environmental Science and Ecotechnology 30 (2026) 100684

Contents lists available at ScienceDirect

Environmental Science and Ecotechnology

journal homepage: www.journals.elsevier.com/environmental-science-and-ecotechnology/

Original Research

Green prompt engineering for sustainable generative AI

Sanjay Podder *, Hema Date, Shankar Murthy

Indian Institute of Management Mumbai, Powai, Mumbai, Maharashtra, 400087, India

ARTICLE INFO

Article history:
Received 15 July 2025
Received in revised form
12 March 2026
Accepted 12 March 2026

Keywords:
Green computing
Greenhouse gases
Generative AI
Inference mechanisms
Prompt engineering
Green AI

ABSTRACT

Prompt engineering involves manual design and optimization of text-based instructions or queries, enabling precise control over outputs generated by pre-trained large language models (LLMs) and ensuring alignment with desired responses. However, substantial computational costs and energy footprint of prompt inferencing process remain critical challenges while building generative AI applications. The energy efficiency of LLM inferences is particularly impacted by suboptimal prompts, which may require multiple iterations, thereby escalating energy consumption and the associated carbon footprint. To address these challenges, we propose a series of practices and guidelines designed to enhance the likelihood of obtaining desired responses from LLMs with minimal reiterations. Empirical evaluation demonstrates that, across a range of LLMs and test scenarios, energy consumption and corresponding operational greenhouse gas emissions were reduced by 32–48% when best practices were applied. Drawing upon these insights, our proposed best practices can be seamlessly integrated into the design frameworks of generative AI applications, thereby enhancing the energy efficiency of prompt inferencing. By addressing the challenge of establishing a cohesive framework for energy-efficient prompt design and inferencing, this paper advocates for the sustainable and effective deployment of generative AI technologies.

1. Introduction

In recent times, there has been a new wave of generative artificial intelligence (AI) systems aimed at producing novel content, such as text, code, images, audio, and other types of data, based on the patterns and relationships present in the data on which they were trained [1]. These systems, due to their ability to generate human-like text, understand complex queries, and perform intricate tasks, are increasingly being adopted across various industries, from content creation and customer service to research and development [2].

Prompt engineering, a crucial component of generative AI, involves designing and optimizing prompts to improve the quality and relevance of content generated by pre-trained large language models (LLMs) Mundlamuri. [3,4], such as ChatGPT. A prompt is a text-based instruction or query that is given to an LLM to generate the desired response [5]. Prompt engineering involves controlling the output of generative AI models to ensure they produce the expected results by harnessing their full potential and making them more accessible and applicable across diverse domains. The importance of prompt engineering is underscored by its ability to guide model responses, thereby enhancing the versatility and relevance of LLMs across sectors [6].

However, designing prompts is an experience and skill-driven manual activity that presents multiple challenges to prompt engineers and users while drafting optimal prompts.

1.1. Challenges in prompt engineering

Lack of guidance in trial and error. Little systematic guidance exists for designing computationally efficient and energy-optimal prompts, and prompt engineering is largely a trial-and-error process [7–9].

Computational costs. Despite recent advances in building small language models, such as Llama 3.2-1B and DeepSeek-R1-1.5B [10], the widespread use of large LLMs entails significant computational costs and delays in prompt refinement [7].

Contextual intent. It is often difficult to identify the full user intent underlying prompts, thereby limiting the ability to generate

This article is part of a special issue entitled: Green AI and Sustainable Computing published in Environmental Science and Ecotechnology.

* Corresponding author.
E-mail address: Sanjay.podder.2116009@iimmumbai.ac.in (S. Podder).

https://doi.org/10.1016/j.ese.2026.100684

contextually relevant and accurate responses [11].

Stability and robustness of recommendations and inference. LLM responses are often sensitive to the details of the prompt wording and to ongoing model revisions, including fine-tuning.

Testing and debugging. Prompt engineers must test and debug their solutions to ensure they function as expected, a process that can be complex and time-consuming [12].

Interdisciplinary knowledge. A blend of technical understanding and domain-specific knowledge is often required to build prompt-driven generative AI applications.

Real-time adaptation. Adapting prompts in real-time to changing contexts or user inputs is a complex task. Ensuring that generative AI systems can dynamically adjust their responses based on evolving scenarios adds another layer of complexity.

Prompt inefficiencies. Ineffective prompts can lead to misaligned outputs, requiring multiple iterations and increasing energy consumption due to repeated inferencing. Prompt inefficiencies directly translate to higher energy consumption during LLM inference by increasing inference latency and token counts, both of which are positively correlated with power draw on the graphical processing unit (GPU)-based hardware scaling linearly (Pearson $r \approx 0.996$) across multiple models and tasks) [13]. Suboptimal prompts can result in up to a twofold increase in per-query energy usage under worst-case conditions [14,15]. Even marginal inefficiencies, such as a few redundant tokens per prompt, can lead to substantial additional energy consumption at scale. Caravaca et al. [16] highlighted that excessively long inputs may exceed the GPU's key-value (KV) cache capacity, which leads to slower memory operations and causes disproportionate spikes in energy usage. Moreover, ambiguity in prompts introduces additional inefficiency, as models may generate longer responses to address multiple interpretations or require user clarification, thereby increasing overall energy usage. Empirical evidence demonstrates that when structural inefficiencies are eliminated by optimizing prompts with custom tags, energy consumption can be reduced by 7% for zero-shot (prompt without any example), 99% for one-shot (prompt with one example), and 83% for few-shot (prompt with multiple examples) paradigms in Llama 3 code tasks [15].

There is an increasing number of recent studies looking at green AI and efficient LLM inference (Table 1). The next section discusses how prompt engineering can contribute to reducing the environmental impact of generative AI.

1.2. Energy consumption and carbon emissions of prompt inferencing

According to recent estimates [23], by 2028, AI-specific electricity demand is projected to rise to 165–326 TWh per year, exceeding the total current electricity use of all data centers in the United States. The inferencing process accounts for 80–90% of AI's computing power usage because it is inherently resource-intensive, particularly when executed continuously at very large scales [19,24,25]. For example, according to Chien et al. [19], OpenAI handles over 270 million prompt inference requests daily, with each prompt consisting of 1200 tokens on average [26]. The energy consumption and associated carbon emissions of prompt inferencing are substantial (both Scope 2 and Scope 3) and are expected to increase further as the adoption of generative AI applications grows. For instance, GPT-4's carbon emissions from just 121 days of inference are estimated to be equivalent to its entire training emissions [27]. Similarly, in a blog on Smartly.AI [28], the authors estimated that each prompt inference results in ~4.3 g of CO_2eq.

Generative LLM inference for a single prompt consists of several forward passes, generating each output token sequentially. Initially, all input tokens in a prompt are processed in a forward pass to generate the first output token (the prefill phase). Then, in the decode phase, the next set of output tokens is generated as per the forward pass of the last token and the cached context from previous tokens. These two phases tend to be computationally and memory-intensive, respectively [26]. For example, a recent empirical study [25] reported that, for the Llama-3 65B model, energy consumption reached 1000 W per second while running inference on batches of 512 prompts across 32 shards per GPU on an NVIDIA V100. Furthermore, for the same configuration, the energy consumption per output token was estimated to be approximately 6 J.

Prompt engineering can have a direct impact on energy use during inference, as it guides LLMs as to what kind of response needs to be generated. However, energy consumption and associated carbon emissions during inference vary significantly with several factors, including model size, the hardware used, and the efficiency of the underlying systems that support LLMs [21,29–31]. Model size, typically measured by the number of parameters, is a primary determinant. Larger models inherently require more computational power and memory bandwidth, thereby increasing energy consumption [25]. The choice of hardware, for example, the specific GPU architecture chosen (e.g., A100, V100, or H100), can influence energy usage, with newer generations often offering improved performance per watt [21,30]. Furthermore, the way in which these models are used, such as the length and complexity of prompts and the batch size of inference requests, can influence overall energy consumption. Finally, the LLM's underlying architecture is a significant factor in its energy efficiency [19]. While transformer-based models are currently dominant, variations in architecture, such as the number of layers and the types of attention mechanisms employed, can affect energy consumption. Emerging alternative architectures, such as receptance-weighted key value (RWKV) [32] and Megalodon [33], aim to provide more efficient methods for LLM inference.

We listed some factors that affect the energy use and carbon impact of prompt inference, along with a description of how prompt engineering can potentially influence this impact:

Inference time. By carefully crafting prompts, prompt engineers can guide LLMs to provide the desired outputs more efficiently. Well-designed prompts can help LLMs find the desired response more quickly, thereby reducing overall inference time. Since energy consumption is strongly correlated with latency, shorter inference times can reduce energy consumption [28].

Computational resources. LLMs often require substantial computational resources during inference, and resource allocation can affect energy consumption [14,34]. Prompt engineering can optimize the utilization of computational resources by providing specific instructions, constraints, or contexts, thereby reducing overall computational load and, consequently, energy consumption [13,34].

Model size. The size of the language model can affect energy consumption during inference [16,25]. By designing prompts that focus on specific aspects of the task or domain, one can limit the portion of the model used. This approach can be beneficial in scenarios in which a smaller, specialized model yields sufficient results, thereby reducing computational requirements and energy consumption [35,36].

Prompt characteristics. The characteristics of the prompt and its response, such as their length and complexity, also influence energy consumption. Longer prompts require the model to process more tokens, increasing the computational workload [37] and thus higher energy consumption during the prefill phase [13]. Moreover, the length of the generated response can impact energy usage, as generating longer text requires additional computation

Table 1
Recent works on green AI and efficient LLM inference (2023–2025). Key contributions and relevance to prompt efficiency and sustainability.

Work	Focus and findings	Relevance	Reference
Eco-friendly digital twin	Introduces Green AI principles in practice; proposes a security surveillance framework that reuses low-resource devices to save energy. Emphasizes shifting from "Red AI" (performance-only) to "Green AI" (including environmental costs).	Highlights the broader Green AI field and demonstrates energy-conscious design in AI systems, reinforcing the need to consider efficiency (in our case, via prompt design) alongside performance.	[17]
AI at Google scale	Measures real-world AI inference impact in production (Google's Gemini). Finds median text prompt uses only ~0.24 Wh with optimizations. Achieved 33 × energy and 44 × carbon reduction in one year through software efficiency and 90% carbon-free energy.	Provides industry-scale evidence that inference efficiency can be vastly improved. Underscores the importance of optimizing the "full stack"—our prompt-level optimizations complement such system-level gains by further reducing per-query computation.	[18]
CarbonMin scheduling	Proposes CarbonMin, an algorithm to route LLM requests to datacenters with cleaner power. Maintains user latency while cutting inference emissions by ~35% (today's grid) and projected ~56% (2035 grid).	Illustrates a deployment-time strategy for Green AI. Contextualizes our work: even after a prompt is crafted, when and where it has executed matters. In a complete solution, prompt efficiency and carbon-aware scheduling would work in tandem to minimize emissions.	[19]
Prompt engineering & energy	Empirical study of how various prompting techniques (zero-shot, few-shot, chain-of-thought) affect energy usage for code generation. Suggests that using special prompt tags/structure can reduce LLM inference energy without hurting accuracy.	Confirms that prompt design choices can impact energy. Validates the concept of prompt-level energy optimization. Our work expands on this by evaluating new prompt strategies and quantifying their energy or CO_2 savings across different tasks.	[15]
Green prompting	Investigates prompt/response features affecting energy across multiple tasks. Finds that semantic complexity and certain keywords in prompts drive energy use more than raw length. Recommends simplifying prompt intent to essentials to save energy.	Supports our guidelines to avoid unnecessary complexity and ambiguity in prompts. Provides evidence that "lean" prompts (with clear intent) are an important lever for Green AI. We incorporate this insight by stressing semantic clarity.	[20]
LLM energy benchmark	Benchmarks energy for various LLMs and tasks. Reports: (1) Linear scaling of energy with output length; (2) Strong correlation between latency and energy; (3) No accuracy drop when running models at lower precision to save power.	Reinforces best practices: limiting generated output length can directly cut energy and designing prompts that yield faster convergence saves energy. Suggests system optimizations could pair with prompt optimizations for additive benefits.	[13]
Hardware/ architectural advances	Surveys and develops methods to make LLM inference more efficient: model compression, hardware acceleration (GPUs, TPUs), and new architectures (Transformer variants like Megalodon, recurrent models like RWKV).	These works address the problem from the model and hardware perspective, whereas we focus on the prompt design. Our prompt guidelines are model-agnostic and can boost efficiency on any architecture.	[21,22]

Abbreviations: LLM, large language model; GPU, graphical processing unit; TPU, tensor processing unit; RWKV, receptance-weighted key value.

[13]. By controlling the desired output length through prompts, one can influence the model to generate more concise responses, potentially reducing energy consumption. Similarly, more complex prompts can involve intricate semantic relationships, thereby requiring more computational resources for the model to process. Recent empirical studies have shown that, even when presented with identical tasks, variations in prompt design can lead to distinct energy consumption patterns among underlying LLMs [20]. This highlights the potential of prompt engineering as a software-level optimization strategy that can be implemented without modifying the LLM architecture or requiring specialized hardware.

In this paper, we consider several principles (or best practices) of green prompt engineering, providing a set of guidelines for designing prompts that reduce LLM workload when eliciting expected responses by activating them as minimally as possible. These practices are discussed in section 3, along with an empirical analysis of potential energy savings across various use cases.

1.3. Related works

While numerous sources provide comprehensive guidelines for improving prompt construction [25,38–40], the intersection of prompt engineering and sustainability remains underexplored. This is the case despite numerous studies on improving energy-efficient inference mechanisms in deep learning [19,26,30,33,34] and on benchmarking energy consumption during prompt inference in LLMs [13,14,29]. There still exists a significant lack of research that rigorously evaluates and quantifies the energy consumption and carbon emissions associated with specific prompt design principles. This research seeks to address this critical gap by systematically analyzing the environmental impact of various prompt engineering design practices. By leveraging advanced metrics and assessment techniques, this study aims to delineate the sustainability dimensions inherent in prompt engineering, thereby contributing to the broader discourse on optimizing both the performance and carbon footprint of large-scale generative AI system deployments.

Recent literature has begun to explore the broader context of green AI and its implications for inference efficiency [41]. Kim and Ben-Othman [17] proposed an eco-friendly low-resource security surveillance framework for digital twin services, emphasizing the importance of incorporating environmental, economic, and social costs into intelligent systems. Their work highlights the shift from performance-centric "Red AI" to sustainability-aware "Green AI," a perspective increasingly relevant in smart city applications and beyond.

At the infrastructure level, Elsworth et al. [18] conducted a large-scale study at Google to measure the environmental impact of delivering AI services. They measured the real-world energy usage of AI assistant prompts and found that a median text query consumes ~0.24 Wh under Google's optimized infrastructure. The authors mentioned that through software efficiency improvements and clean energy procurement, Google achieved a 33 × reduction in energy consumption and a 44 × reduction in the carbon footprint over one year. These results underscore the potential of system-level optimizations to reduce the environmental impact of AI inference. Complementing these efforts, Chien et al. [19] introduced CarbonMin, a carbon-aware scheduling algorithm that routes inference requests to data centers with lower carbon intensity. Their study demonstrated that such geographic shifting could reduce emissions by 35% under current grid conditions and

by up to 56% by 2035, with further reductions possible as data center headroom increases. These deployment strategies highlight the importance of not only how prompts are designed but also where and when they are executed.

In Rubei et al. [15], the authors analyzed and compared the energy consumption associated with different prompt engineering techniques (PETs), such as zero-shot prompting, one-shot prompting, multi-shot prompting, and chain-of-thought prompting, when applied to code completion tasks for Java using LLaMA 3.1 [42]. In this paper, in contrast to Rubei [15], we propose novel prompting guidelines that can be applied in conjunction with any of these PETs, together with an analysis of the expected reductions in energy savings and associated greenhouse gas (GHG) emissions.

Another recent study [20] focused specifically on how different prompt and response characteristics directly impact the energy cost of LLM inference. One of the key findings of this study was the significance of semantic meaning in determining energy consumption. The research demonstrated that the underlying meaning and complexity of the task conveyed in the prompt have a more substantial influence on the energy used during inference than the number of tokens in the prompt itself. This suggests that optimizing prompts for clarity and conciseness in their semantic intent is more crucial for energy efficiency than simply reducing word count. Another important result from this study was that specific keywords (specific to the task) were found to be associated with the level of energy usage during inference. This implies that the vocabulary chosen for prompts can subtly affect the computational pathways within an LLM, leading to variations in energy consumption.

The green prompting study by Adamska et al. [20] further supports these findings by showing that semantic complexity and certain domain-specific keywords significantly influence energy consumption. The authors recommended simplifying prompt intent and avoiding unnecessary verbosity to reduce energy usage, which aligns closely with our proposed best practices. Poddar et al. [13] offered further relevant insights through a comprehensive benchmarking of LLM inference energy for a range of Natural Language Processing (NLP) tasks across different LLM families and sizes. The study revealed a strong positive correlation between response time and inference energy for locally run open-source LLMs. This suggests that response time can serve as a reliable proxy for estimating energy consumption in such setups, potentially obviating the need for specialized energy measurement tools. In addition, the relationship between energy usage and both input and output sizes was found to be linear, despite the theoretical expectation of a quadratic dependence on input size due to the self-attention mechanism. This linearity was attributable to hardware optimizations, such as parallel processing and caching. The slope of the energy increase was steeper for output length compared to input length, indicating that the process of generating output tokens is more energy-intensive than processing the input. This is primarily due to the autoregressive nature of output generation, which lacks the parallelism achievable during input processing.

In addition to prompt-level strategies, recent research has focused on improving the energy efficiency of LLM architecture and hardware. Kachris [21] and Li et al. [30] surveyed specialized AI hardware accelerators and techniques, such as quantization, batching, and mixed precision, which significantly reduce energy consumption during inference. Upgrading from older GPU architectures (e.g., NVIDIA V100) to newer ones, such as A100 and H100, yields markedly higher performance per watt [21]. Moreover, emerging architectures, such as RWKV [32] and Megalodon [33], aim to process long contexts more efficiently, potentially altering how prompt length and structure affect energy usage. Speculative decoding is another emerging research direction [22] aimed at accelerating LLM inference by using a smaller draft model to propose multiple tokens at once, which the main LLM model then verifies, thereby achieving 2–4 times speedup without changing results, making it valuable for real-time LLM use.

In summary, prior research has clearly determined that inference is a major contributor to AI's environmental footprint (often outweighing training [19]) and that mitigating this impact requires efforts on multiple fronts—from greener infrastructure to smarter model design and usage. While these studies are orthogonal to our work, they provide a valuable context. Our approach to optimizing prompts can be seen as model-agnostic software optimization that can further enhance the gains from efficient hardware or model designs. By building on insights from recent work and addressing gaps in prompt-focused energy analysis, our study aims to advance understanding of how careful prompt design can serve as an effective lever for reducing LLM inference emissions. We extend the literature by providing empirical evidence of the energy and GHG reductions achievable through concrete prompt guidelines, thereby contributing a new dimension to the green AI discourse. In practice, achieving sustainable AI requires both improving the underlying model/hardware efficiency and refining the prompting strategy to minimize avoidable computation.

1.4. Research questions

In light of the previously identified research gaps, this study aims to rigorously examine the energy and carbon implications of prompt design practices by addressing the following research questions:

Q1: What quantifiable reductions in energy consumption and associated GHG emissions can be achieved through the adoption of energy-efficient prompt design practices?
Q2: Which specific guidelines should be adhered to during the design of prompts to minimize the energy consumption and carbon footprint associated with the inference process while still ensuring accurate and reliable model outputs?

2. Experimental design

To experimentally assess the energy savings achievable by following best practices, we considered three LLMs: GPT-4o [36], Cohere/Command R + [20], and Mistral-7B [43]. These models were selected to represent a variety of both sizes and types—specifically, proprietary versus open-source options—so that our experimental conclusions would be applicable across different LLM architectures.

GPT-4o, for instance, can have approximately 200 billion parameters, which refers to the number of tunable weights or connections in the neural network that the model uses to generate text [17]. This high parameter count correlates with greater computational complexity and, consequently, higher energy consumption. In contrast, Cohere/Command R+ utilizes 104 billion parameters, making it a moderately sized LLM. Mistral-7B is a much smaller model, with 7 billion parameters. By including models of varying sizes, we aimed to examine how energy consumption scales with model complexity and to ensure that our findings are applicable to both large, powerful LLMs and smaller, more resource-efficient ones.

In terms of access, GPT-4o and Cohere/Command R+ were interfaced through their respective application programming interfaces (APIs), which means that we sent prompts to the cloud-hosted versions of these models. On the other hand, Mistral-7B

was locally installed; therefore, all computations for this model were performed on-site, providing direct control over the hardware and measurement environment.

To estimate the energy consumption associated with running these models, we used the EcoLogits Calculator [44] via its API. Ecologits is an open-source tool designed to estimate the energy consumption and environmental impacts (e.g., GHG emissions) associated with the use of generative AI models. It operates by leveraging the life cycle assessment (LCA) methodology, as outlined in the ISO 14044 standard [45]. LCA is a comprehensive method for evaluating the environmental impacts of a product or service throughout its entire life cycle, from raw material extraction to disposal. By applying this method, Ecologits provides a more holistic view of the environmental footprint of AI inference calls. The tool supports a range of LLM providers, including well-known names, such as OpenAI, Anthropic, and Mistral AI. Additionally, EcoLogits offers an online calculator that facilitates the estimation of environmental impacts associated with LLM inference, including energy consumption and GHG emissions. For the purpose of GHG emissions calculations in our experiments, we assumed that all LLMs were operating within the United States and used that country's average electricity mix (conversion factor) of 0.68 kg CO_2eq per kilowatt-hour (kWh), which is a standardized metric for estimating the amount of carbon dioxide-equivalent emissions generated per unit of electricity in the US. By incorporating these assumptions, we aimed to provide realistic, comparable estimates of the environmental impact associated with using each LLM in our tests. Bespoke prompt test dataset tailored to our study was created recognizing the absence of publicly available benchmark prompt datasets annotated with precise energy-consumption values and the inherent challenges of retrofitting such datasets to our specific experimental requirements. This dataset was meticulously curated to encompass 103 real-world application scenarios across best practices (BPs), spanning multiple domains, including software engineering, technical content generation, travel consultancy, and financial planning. The selection criteria were designed to maximize representativeness and generalizability, ensuring that the evaluated prompts reflected the diversity of tasks encountered in the practical deployments of LLMs.

For each scenario within the dataset, we systematically devised two distinct sets of prompts. The first set consisted of baseline prompts crafted without deliberate adherence to established BPs in prompt engineering. These prompts frequently necessitated iterative refinement and multiple interactions with the model to achieve satisfactory outputs, thereby simulating suboptimal real-world usage patterns. In contrast, the second set of prompts was designed to address the same scenario while explicitly incorporating recommended BPs. These practices include, but are not limited to, precise instruction formulation, clear context provision, and strategic constraint setting—techniques that have been empirically shown to enhance model performance and efficiency.

By structuring the dataset in this manner, we enabled a controlled comparative analysis of energy consumption and computational overhead associated with BP versus non-BP prompting paradigms. This approach enabled rigorous quantification of the potential energy savings and environmental benefits achievable through disciplined, timely engineering, while also illuminating the operational trade-offs relevant to expert practitioners and researchers in the field.

The following is an overview of the problem contexts considered:

- As a technical writer, a user seeks to draft a brief article on prompt engineering.
- As a content marketer, a user requires captions for an Instagram post about a sustainability campaign.
- As a historian, a user seeks key points on the Renaissance for a presentation.
- As a manager, a user needs a brief professional team email summarizing project updates.
- As a small business owner, a user wants a brief, concise description for a product listing.
- A user seeks details on books and cars in the eXtensible Markup Language (XML) format.
- A user is preparing for a trivia quiz and requires concise bullet-point answers.
- As a Python learner, a user requests clean, commented code examples.
- As a data analyst, a user requires JavaScript Object Notation (JSON) output to parse details about trending movies.

During the construction of our test prompt set, we intentionally excluded problem scenarios in which a direct query via Google Search would yield an unambiguous and satisfactory answer. This exclusion criterion was crucial to ensure that the evaluation focused on tasks requiring advanced reasoning and generative capabilities, rather than mere information retrieval. Each BP test scenario was executed in a newly initialized session to eliminate context carryover or session-based learning effects, thereby ensuring that each response from the language model was uninfluenced by prior prompts or responses. Furthermore, all API outputs were systematically saved to enable post hoc analysis and minimize redundant API calls, which is essential for experimental consistency.

For each BP, we computed the per-scenario percentage reductions in energy consumption and GHG emissions across all three LLMs. Margins of error were derived using a two-tailed 95% confidence interval (CI) over the mean reduction, assuming a *t*-distribution due to a finite number of test scenarios. No weighting was applied across the models, and each scenario contributed equally. The reported confidence intervals correspond to the upper and lower bounds of the CI for the mean percentage reduction.

In the following sections, we provide an in-depth examination of the experimental procedures and outcomes related to each identified BP. Subsequently, we synthesize an overall analysis that compares the effectiveness of these BPs within and across LLMs, highlighting the patterns, strengths, and limitations observed in our evaluation.

3. Green prompt engineering best practices

In this section, we discuss various BPs for drafting prompts to reduce energy consumption and GHG emissions in generative AI applications. To assess the expected energy savings and GHG emissions reductions for each BP, we conducted experiments and measured energy consumption.

3.1. Best practice #1: variational prompting

Prompting refers to providing queries or instructions to an LLM to generate the desired response. Certain scenarios require prompt engineers to send multiple prompts to an LLM to fetch the required output. For instance, a user requires some resolution to a problem or needs to perform early-stage activities, such as gathering information, exploring different possibilities, or evaluating potential options. Such scenarios lead to multiple prompt iterations to the LLM, which can lead to energy inefficiency, since users might require multiple solutions.

Therefore, prompting LLMs to generate different variations at

once in their responses can reduce prompt iterations because one of the variations is likely to be the response that meets the expectation; thus, repeated prompts need not be generated to get this response. This method is suitable for exploratory use cases in which the user is researching or seeking solutions to a problem. This method is more effective when prompts are specific rather than generic.

The underlying mechanism for this BP stems from the LLM's ability to explore a broader solution space within a single inference pass. By generating multiple potential answers simultaneously, the need for subsequent prompt reiterations or separate inference calls to explore alternative solutions can be minimized, thereby reducing overall computational work and energy expenditure. This method can be implemented by following the guidelines outlined below.

3.1.1. Guideline 1: ask for variations while prompting when you want to find the best way to solve a problem

Example:

- *Scenario*: As a Java developer, a user wants to find efficient methods to sort arrays in Java.
- *Original prompts*:
  - o Prompt 1: How can an array be sorted in Java?
  - o Prompt 2: Show me other ways to sort an array.
  - o Prompt 3: Keep only the efficient methods from the above responses.
- *Recommended prompt*: Show me different efficient ways to sort an array in Java?
- *Experimental observations*: With the original Prompt 1, the LLMs responded with the commonly used method to sort an array, i.e., array.sort(). Only after refining the prompts a further two times was the LLM able to finalize a list of efficient methods. However, with the recommended prompt, the LLM responded with various efficient ways of sorting arrays in Java apart from array.sort(), such as parallelSort() and collections.sort().

3.1.2. Guideline 2: specific phrases in prompts, such as "effective ways," "solutions," "methods," and "techniques," make LLMs respond with deeper details

Example:

- *Scenario*: As an AI developer, a user wants to know the methods with which to perform data distillation before training AI models.
- *Original prompts*:
  - o Prompt 1: I need to train an AI model. How can I perform data distillation before training my model?
  - o Prompt 2: Please suggest methods, not steps.
- *Recommended prompt*: I need to train an AI model. How can I perform data distillation before training my model? Suggest the main methods to distill data.
- *Experimental observations*: With the original prompt, when asked how to perform data distillation before training an AI model, the LLM responded with the steps for data distillation. However, the user did not ask explicitly for the steps. Therefore, by modifying the prompt using the method variant, the user was able to obtain the desired result using various methods for data distillation, such as random sampling, stratified sampling, active learning, transfer learning, and clustering.

Experimental analysis. Variational prompting approaches yield measurable differences in energy consumption and associated GHG emissions. An experimental analysis of 22 use-case scenarios modeled using 71 prompts supports this observation (Table 2).

All three models show notable reductions in both energy consumption and GHG emissions following the application of this BP. The reductions ranged from 39% to 50%, indicating the effectiveness of the BP. GPT-4o had the highest baseline energy and emissions but also achieved the largest absolute and percentage reduction (50%). Cohere/Command R+ showed a moderate footprint and a 44% reduction, balancing performance and efficiency. Mistral-7B, even with the lowest energy and emissions, still had a 39% reduction.

However, in a small number of scenarios (9% of the cases), we observed an increase in energy consumption and GHG emissions (>50%). This negative impact resulted primarily from the fact that these scenarios did not require a broader exploration of alternatives. That is, there was one optimal answer, which was generated even without applying the BP (in other words, the application of the BP resulted in redundant answers, thus increasing inference energy consumption as well as associated GHG emissions).

Across all three models, variational prompting yielded an average 36% reduction in energy consumption and associated GHG emissions, with an 11% margin of error (CI: 25–47%). The findings highlight the effectiveness of variational prompting in reducing emissions in cases suitable for its application, but they also reveal that, in scenarios where it may be ineffective, its use could significantly increase emissions. This observation indicates the need for further optimization to ensure that variational prompting is applied judiciously and effectively across diverse scenarios.

Tradeoffs. LLMs often provide expected responses even without applying BPs if the problem scenario has few optimal answers. In such cases, the LLM's response while using this method will be a slight upgrade to the original prompt or might even result in redundant variations, thus increasing energy consumption.

3.2. Best practice #2: avoiding local lingua in prompts

Local lingua refers to language or jargon that is specific to a particular context, such as a project, group, or domain, which the LLM might not have encountered during its training. Such local terminology, although it might be "well understood" by the prompt designer while drafting prompts, might be uninterpretable to the LLM. Unless an LLM has been retrained on a corpus containing local lingua (e.g., project-specific terminology), using local lingua in prompts can elicit incorrect responses and can result in avoidable re-prompting, thus causing energy inefficiency. Greater use of the local language was associated with a higher likelihood of inaccurate LLM responses.

Therefore, when writing a prompt, it is important to explicitly identify non-standard terminology, such as project-specific jargon, and replace this with standard terms or else supplement the prompts with the meanings of such terms. In scenarios where users need to use project-specific terminology across multiple prompts, it is good practice to provide the context for these terms and their standard forms through in-context learning. This will help LLMs understand the exact requirements and generate the expected responses.

A potential reason lies in the reduced complexity of LLM prompts. When the language used is universal, LLMs require less processing to interpret a prompt's intent, potentially leading to more direct and concise responses, which, in turn, reduces the output token length and the associated energy cost. As highlighted in a recent study [20], the significance of semantic meaning underscores the importance of clarity in prompt language for efficient processing. This method can be implemented by following

Table 2
Energy and carbon impact of variational prompting.

LLM	Average energy without applying BP (Wh per scenario)	Average energy after applying BP (Wh per scenario)	Average GHG emission without applying BP (g CO_2eq per scenario)	Average GHG emission after applying BP (g CO_2eq per scenario)	Mean of energy and GHG emission reduction
GPT-4o	117.3	59.0	82.0	41.3	50%
Cohere/ Command R+	22.2	11.6	15.5	8.7	44%
Mistral-7B	9.1	5.5	6.2	3.8	39%

Abbreviations: LLM, large language model; GHG, greenhouse gas; g CO_2eq, gram CO_2 equivalent; BP, best practice.

the guidelines outlined below.

3.2.1. Guideline 1: validate prompts via readability tools and replace project-specific terminology with generic terms

Example:

- *Scenario*: A user wants to ask a colleague for his/her bandwidth to test a web page. However, before seeking assistance, the user receives guidance from the LLM on how long it might take.
- *Original prompt*: How much bandwidth is required of the testing team to test a web page with 10 features?
- *Recommended prompt*: How much capacity is required of the testing team to test a web page with 10 features?
- *Experimental observations*: With the original prompt, the LLM responded with network bandwidth details, whereas the user sought information on the team's capacity. Therefore, when the recommended prompt was provided, the appropriate answer was received, as expected by the prompt engineer.

3.2.2. Guideline 2: provide proper context, details, and explanations for the local lingua terms used in the prompts rather than vague outputs to achieve the desired results

Example:

- *Scenario*: As a business analyst, a user wants to create a user story for creating an insurance policy for a customer with his/her spouse, such as the existing user story CRS28.
- *Original prompt*: Create a user story for the "customer insurance with spouse," such as user story CRS28, including details for the type of data to be provided, such as the policy amount, coverage%, coverage option, and acceptance criteria.
- *Recommended prompt*: Create a user story for the "customer insurance with spouse," such as user story CRS28: "As a customer, I want to create an insurance policy with my son as a beneficiary." Include details for the type of data to be provided, such as policy amount, coverage%, coverage option, and acceptance criteria.
- *Experimental observations*: With the original prompt, the LLM responded with a random scenario, contrary to the prompt engineer's requirement. It generated the user story: "As a customer, I want to have an integrated view of both my insurance policy and my spouse's policy." However, the user wanted a user story, such as: "As a customer, I want to create an insurance policy with my spouse as a beneficiary." Therefore, providing proper context along with the local lingua can yield the desired response.

Experimental analysis. We summarized the average energy and GHG emissions estimates across the three models accessed through their APIs (Table 3). We considered 18 use-case scenarios, modeled through 56 prompts.

All three models showed clear reductions in both energy use and GHG emissions after applying the BP. The percentage reduction varied significantly, with Cohere/Command R+ achieving the highest efficiency gain (45%). GPT-4o, despite being the most powerful model, showed the smallest relative reduction (27%), although its absolute energy and emissions values were the highest. Cohere/Command R+ demonstrated a strong balance between performance and environmental efficiency, with nearly half of the emissions and energy use after applying the BP. Mistral-7B started with the lowest baseline and still achieved a 41% reduction.

In contrast to the BP1, the increase in energy and GHG emissions for this BP was just 2% (on average across all three models) for 11% of the test scenarios.

Across all three models, avoiding local lingua in prompts resulted in an average energy savings of 37.7% and a corresponding reduction in associated GHG emissions, with a 9% margin of error (CI: 28.7–46.7%).

The findings highlight the effectiveness of this BP in reducing emissions across a wide range of scenarios.

Tradeoffs. In experiments, it was observed that LLMs may provide expected responses even when local lingua is present in the prompt when the local lingua terms are not critical to generating correct responses.

Example:

- *Prompt*: "Write the details to be added to the defect document for the defect 138: Jenkins build queue freezes even when build agents are available."
- *Experimental observations*: The LLM responded with the correct defect details for the new defect, including everything that should be part of a defect document, although the reference to the existing defect was ignored. A defect document is a common document that includes similar details. Hence, the LLM responded with the appropriate answer based on the new defect description.

3.3. Best practice #3: avoiding terminological ambiguities

Terminological ambiguity refers to the uncertainty that arises when a particular term or phrase has multiple meanings or interpretations. Using such terms or phrases without proper context in prompts may lead an LLM to respond with incorrect details, necessitating multiple iterations of prompting and resulting in energy inefficiency. The higher the use of terminologically ambiguous prompts, the greater the chance of multiple iterations.

Therefore, prompt engineering should use precise and unambiguous terminology (e.g., full forms rather than abbreviations or acronyms) or add the context for the terminologically ambiguous phrases to avoid unintended interpretations of these terms. By using clear, complete terminology and appropriate context, Prompt engineers provide the LLM with a precise request, enabling

Table 3
Energy and carbon impact of avoiding local lingua in prompts.

LLM	Average energy without applying BP (Wh per scenario)	Average energy after applying BP (Wh per scenario)	Average GHG emission without applying BP (g CO_2eq per scenario)	Average GHG emission after applying BP (g CO_2eq per scenario)	Mean of energy and GHG emission reduction
GPT-4o	83.8	61.4	58.7	43.0	27%
Cohere/ Command R+	24.7	13.7	17.3	9.6	45%
Mistral-7B	11.9	7.0	8.3	4.9	41%

Abbreviations: LLM, large language model; GHG, greenhouse gas; g CO_2eq, gram CO_2 equivalent; BP, best practice.

it to respond with correct output in a single prompt and avoiding prompt reiteration, thereby improving energy efficiency. This BP reduces processing overhead and the likelihood of generating irrelevant or lengthy responses due to misinterpretations, thereby minimizing the need for clarification and subsequent prompt iterations.

This method can be implemented by following the guidelines outlined below.

3.3.1. Guideline: use glossary preambles for domain-specific terms, e.g., full terms for abbreviations or shortened forms

Example:

- *Scenario*: As a developer, a user wants to create a technical design document for a new application.
- *Original prompt*: As a developer, I want to create a TDD for my new web application. Suggest a template for doing this.
- *Recommended prompt*: As a developer, I want to create a technical design document for my new web application. Suggest a template for doing this.
- *Experimental observations*: TDD means different things in different contexts. It can refer to test-driven development, a technical design document, a tentative delivery date in Information Technology (IT), total demand distortion in an electrical system, time-division duplexing in Telecommunications, and so on. With the original prompt, the LLM responded with the test-driven development details, whereas the prompt engineer's intent was to fetch the details to create a technical design document.

Experimental analysis. We summarized the average energy and GHG emissions estimates across the three models accessed through their APIs (Table 4). The experiments included 13 use-case scenarios modeled using 43 prompts.

All three models showed clear reductions in both energy use and GHG emissions after applying the BP with low variation. The percentage reduction varied significantly, with Mistral-7B achieving the largest gain (52%) and GPT-4o the smallest relative reduction (43%). Across all three models, avoiding terminological ambiguities in prompts resulted in an average 48% reduction in energy consumption and associated GHG emissions, with a 13% margin of error (CI: 35–61%). Thus, the findings highlight the effectiveness of this BP in reducing emissions across the use cases.

Tradeoffs. In certain scenarios, LLMs can successfully interpret ambiguous terms when prompts include additional context, particularly in inherently generic domains.

3.4. Best practice #4: role modeling

Providing simple prompts without adding personas does not convey the prompt engineer's clear intentions to LLMs. Hence, the LLM may not generate the expected responses or the in-depth details desired by the prompt engineer, leading to re-prompting and energy inefficiency. By providing a clear role, the prompt guides the LLM to access and utilize a more specific subset of its vast knowledge and generation patterns. This can streamline the response–generation process, reduce the exploration of irrelevant information, and yield more focused and concise answers.

Prompt engineers can follow the guidelines below to implement this method.

3.4.1. Guideline 1: specify a persona in the input to avoid incomplete or incorrect responses and the need for re-prompting

Example:

- *Scenario*: As an emergency response professional, a user wants to draft an email to team members highlighting the achievements and lessons learned from the last quarter.
- *Original prompt*: I want to draft an email to my team members discussing our achievements in the last quarter's results and the learning we can take from these to improve our service in the current quarter.
- *Recommended prompt*: As an emergency response professional, I want to draft an email to my team members discussing our achievements in the last quarter's results and the learning we can take from these to improve our service in the current quarter.
- *Experimental observations*: With the original prompt, the LLM generated a generic email; however, the user wanted the email to be written by an emergency response professional.

Table 4
Energy and carbon impact of avoiding terminological ambiguities.

LLM	Average energy without applying BP (Wh per scenario)	Average energy after applying BP (Wh per scenario)	Average GHG emission without applying BP (g CO_2eq per scenario)	Average GHG emission after applying BP (g CO_2eq per scenario)	Mean of energy and GHG emission reduction
GPT-4o	121.4	69.3	84.9	48.5	43%
Cohere/ Command R+	22.2	11.1	15.5	7.8	50%
Mistral-7B	5.7	2.7	4.0	1.9	52%

Abbreviations: LLM, large language model; GHG, greenhouse gas; g CO_2eq, gram CO_2 equivalent; BP, best practice.

Table 5
Energy and carbon impact of role modeling.

LLM	Average energy without applying BP (Wh per scenario)	Average energy after applying BP (Wh per scenario)	Average GHG emission without applying BP (g CO_2eq per scenario)	Average GHG emission after applying BP (g CO_2eq per scenario)	Mean of energy and GHG emission reduction
GPT-4o	91.8	60.2	64.2	42.1	34%
Cohere/ Command R+	24.9	12.6	17.4	8.8	50%
Mistral-7B	12.9	6.3	8.7	4.4	49%

Abbreviations: LLM, large language model; GHG, greenhouse gas; g CO_2eq, gram CO_2 equivalent; BP, best practice.

Therefore, by providing the persona to the prompts, the user can obtain the desired result in a single prompt, thereby avoiding the need to re-prompt for a specific output.

3.4.2. Guideline 2: including the persona at the beginning enables the LLM to answer all subsequent queries with respect to the role specified in the initial prompt. This reduces the need for re-prompting and improves energy efficiency

Example:

- *Scenario*: A user wants to know what to consider when making big financial decisions about a product or service.
- *Original prompt*: "What do you consider before making a big financial decision about a product or service?"
- *Recommended prompt*: "I want you to act as a top finance executive for a software company who is responsible for making strategic decisions, managing financials, and representing the company to stakeholders. Answer according to the set of scenarios or challenges you face as a professional, using your leadership skills. The first question is 'What do you consider before making a big financial decision about a product or a service?'"
- *Experimental observations:* When the persona is not provided, the LLM responds with generic answers that might be correct in general but might not be suitable if the user is looking for a response related to a particular role. Therefore, it is helpful to provide the proper identity when prompting.

Experimental analysis. We summarized the average energy and GHG emissions estimates across the three models accessed through their APIs (Table 5). It presents an experimental analysis using 61 prompts across 19 distinct use cases.

All three models showed clear reductions in both energy use and GHG emissions after applying the BP. The percentage reduction, however, varied; Cohere/Command R+ and Mistral-7B achieved the highest efficiency gains (~50%). GPT-4o, on the other hand, showed a smaller reduction (34%), although its absolute energy and emissions values were the highest. Across all three models, user modeling in prompts led to an average 44% reduction in energy consumption and associated GHG emissions, with a 15% margin of error (CI: 29–59%).

Tradeoffs. Implementing personas may prove ineffective if other best practices, such as avoiding local lingua and terminological ambiguities, are not adhered to, as this would fail to provide the LLM with a precise and unambiguous description of the intended role or character.

3.5. Best practice #5: specifying optimal response length

The length of the generated response has a strong positive correlation with the energy consumed during inference [13,20]. By explicitly stating the desired length of the expected response, LLMs generate limited yet relevant outputs. Since generating each token involves multiple sequential computations (see Section 1.2), reducing the total number of tokens directly translates into less computational work and lower energy requirements. Additionally, prompt engineers might be required to initiate subsequent prompts to reach the desired length if not specified initially, since truncating prompt responses at certain lengths might render them semantically less relevant.

Therefore, by explicitly specifying the desired response length, prompt engineers can influence the model to generate succinct output and avoid multiple iterations, thereby increasing efficiency. The expected response length can be specified with respect to the count of words or tokens, the number of sentences, paragraphs, bullet points, and so on. This is helpful in scenarios with a word limit, such as writing a post, a blog, or a tweet, or creating presentations with highlighted bullet points.

Prompt engineers can follow the guidelines provided below to implement this method.

3.5.1. Guideline: specify the expected length of the response within the prompt input

Example:

- *Scenario*: As a software testing expert, a user wants to know the differences between Scrum and Kanban.
- *Original prompt*: "Act as a software testing expert. Tell me the difference between Scrum and Kanban."

Table 6
Energy and carbon impact of specifying optimal response length.

LLM	Average energy without applying BP (Wh per scenario)	Average energy after applying BP (Wh per scenario)	Average GHG emission without applying BP (g CO_2eq per scenario)	Average GHG emission after applying BP (g CO_2eq per scenario)	Mean of energy and GHG emission reduction
GPT-4o	46.9	30.9	32.8	21.2	35%
Cohere/ Command R+	8.7	4.4	6.0	3.1	49%
Mistral-7B	6.4	4.4	4.5	3.1	32%

Abbreviations: LLM, large language model; GHG, greenhouse gas; g CO_2eq, gram CO_2 equivalent; BP, best practice.

Table 7
Energy and carbon impact of specifying response format.

LLM	Average energy without applying BP (Wh per scenario)	Average energy after applying BP (Wh per scenario)	Average GHG emission without applying BP (g CO_2eq per scenario)	Average GHG emission after applying BP (g CO_2eq per scenario)	Mean of energy and GHG emission reduction
GPT-4o	66.5	39.7	46.6	27.8	40%
Cohere/ Command R+	14.4	7.1	10.1	5.0	50%
Mistral-7B	12.2	5.2	8.5	3.7	57%

Abbreviations: LLM, large language model; GHG, greenhouse gas; g CO_2eq, gram CO_2 equivalent; BP, best practice.

- *Recommended prompt*: "Act as a software testing expert. Tell me the difference between Scrum and Kanban. Answer in five bullet points."
- *Experimental observations*: With the original prompt, the LLM responded with the details on Scrum and Kanban written in paragraphs, whereas by providing the desired output length in the recommended prompt, the LLM responded in a more structured way by highlighting the key differences in five bullet points between the models. Therefore, giving LLMs the expected output length yields correct results while reducing the number of prompt iterations.

Experimental analysis. We summarized the average energy and GHG emissions estimates across the three models accessed through their APIs (Table 6). It presents an experimental analysis using 57 prompts across 18 use-case scenarios.

Again, all three models showed reductions in both energy use and GHG emissions after applying the BP. The percentage reduction varied, with Cohere/Command R+ achieving the highest efficiency gain (49%). GPT-4o and Mistral-7B, on the other hand, showed smaller reductions. Across all three models, user modeling in prompts yielded an average of 38.7% energy savings and associated GHG emissions reductions, with a 9% margin of error (CI: 29.7–47.7%).

However, for 11% of the test scenarios (2 out of 18), the GHG emissions for the Mistral-7B models increased by 34% after applying the BP, because Prompt engineer specified response length was more than what was needed for a correct response.

The findings highlight the effectiveness of this BP in reducing emissions but also reveal an unexpected increase for some LLMs, suggesting the need for further optimization.

Tradeoffs. LLMs work well with diverse types of response length formats, except for word limits. Currently, not all LLMs are able to answer with the exact number of words specified in a prompt.

3.6. Best practice #6: specifying response format

There are situations in which prompt engineers may expect the generated response to be in a specific format, e.g., for use by a downstream application. Such scenarios might require prompt engineers to send multiple prompts to achieve the desired format, which could lead to energy inefficiency.

By providing a clear blueprint for the response, the prompt can guide the LLM in generating the output more efficiently. This might involve using specific formatting patterns that require less computational overhead than free-form text generation. Furthermore, specifying the format can often lead to more concise and targeted responses, further contributing to energy savings. Therefore, explicitly specifying the desired response style in the input prompt is energy-efficient, as it reduces the need for additional iterations to rewrite the existing output.

This may be helpful even when a particular format style is desired for a set of queries, in which the output style provided in the first prompt applies to all subsequent queries without requiring the format to be specified again. One such application of this method is to generate dummy test data in a particular format.

Prompt engineers can follow the guidelines provided below to implement this method.

3.6.1. Guideline: specify the expected output format in the input prompt, thereby prompting the LLM to respond in the desired style

Example:

- *Scenario*: A user wants details on books and cars, specifically in XML format.
- *Original prompt*: "Can you suggest some good books on health and nutrition?"
- *Recommended prompt*: "Can you suggest some good books on health and nutrition? The response should be in XML format.
- *Example response:*

<books>
<book> <title> Book1 </title> <author> Author1 </author> </book>
<book> <title> Book2 </title> <author> Author2 </author> </book>
</books>

- *Experimental observations*: With the original prompt, the LLM provided a list of books; however, the user requested responses in XML format. Therefore, providing the anticipated format details in the input prompt is beneficial for obtaining correct results and reducing the number of prompt iterations.
- *Subsequent prompt*: "Suggest some good sedan cars within INR 2,000,000 and provide their mileage and fuel capacity."
- *Experimental observations*: The LLM returned the car details in XML format, as prompted by the user with the recommended prompt.

Experimental analysis. We summarized the mean energy and GHG emissions estimates across the three models accessed through their APIs (Table 7). It presents an experimental analysis using 45 prompts across 14 use-case scenarios.

For GPT-4o, the data indicated a 40% reduction in GHG emissions, although one scenario showed a 78% increase. The Mistral-7B model showed a significantly greater reduction of approximately 57%, whereas Cohere/Command R+ showed a 50% reduction, suggesting that the BP may have a different energy impact for different LLMs. Across all three models, variational prompting yielded an average energy savings of 49%, with a 12% margin of error (CI: 37–61%).

Table 8
Energy savings across all best practices (BPs).

Best practice	Average energy without applying BP (Wh per scenario)	Average energy after applying BP (Wh per scenario)	Average GHG emission without applying BP (g CO_2eq per scenario)	Average GHG emission after applying BP (g CO_2eq per scenario)	Mean of energy and GHG emission reduction
BP#1	49.6	23.8	34.6	17.9	52%
BP#2	40.1	27.3	28.1	19.2	32%
BP#3	49.8	28.0	34.8	19.4	44%
BP#4	43.2	26.3	30.7	18.4	39%
BP#5	20.7	13.9	14.5	9.1	33%
BP#6	31.0	16.9	21.7	12.1	45%

Abbreviations: LLM, large language model; GHG, greenhouse gas; g CO_2eq, gram CO_2 equivalent; BP, best practice.

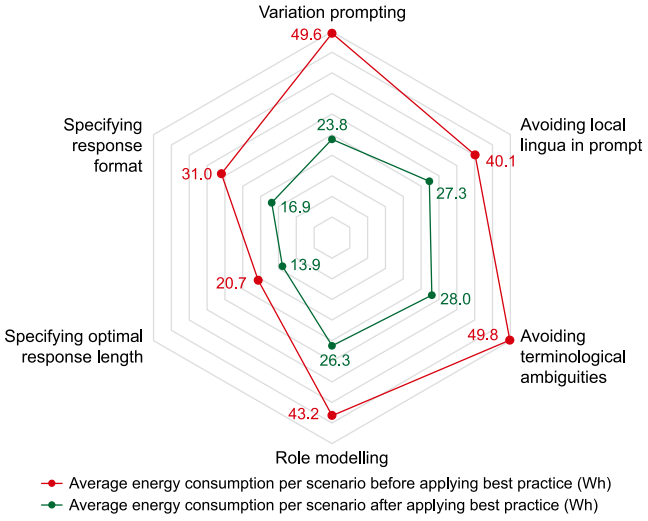

Fig. 1. Spider graph depicting energy savings across best practices.

4. Overall analysis and research findings

Based on experimental analyses of best practices across various test scenarios and LLMs, the following summarizes the empirical findings and addresses the research questions (see Section 1.4).

Average energy and GHG emission reduction estimates for each BP across all three LLMs are summarized across the test scenarios (Table 8, Fig. 1). Whereas BP#1 (variational prompting) and BP#6 (specifying desired response format) showed higher variance in GHG emission reductions across scenarios, BP#5 (specifying optimal response length) and BP#3 (avoiding terminological ambiguities) achieved GHG emission reductions across all test scenarios. This indicates that certain BPs can be implemented with little or no optimization (e.g., BP#5, BP#3, BP#4, and BP#2), whereas others (e.g., BP#1 and BP#6) may be effective in selective applications.

Energy and GHG emission reduction estimates across all three LLMs, averaged across all best practices and covering 100+ use-case scenarios modeled with 300+ prompts, reveal a 39–48% reduction in GHG emissions across the test scenarios (Table 9). Despite substantial variation in actual energy consumption and GHG emissions across these models, due to differences in parameter counts and underlying architectures, the variation in reductions in energy consumption or GHG emissions remains low, indicating the reliability of these data points and associated inferences.

In summary, the application of green prompt engineering BPs demonstrated a tangible positive impact on reducing GHG emissions. The findings reveal a decrease in emissions across all models, ranging from 39% to 48%. However, in a small subset of test scenarios, GHG emissions increased by 5–25%, underscoring the need for further optimization and targeted mitigation strategies. Overall, the adoption of these BPs can yield a 32–52% reduction in emissions, underscoring the potential of prompt engineering to enhance environmental sustainability in generative AI applications.

These prompt engineering techniques align with existing research insights. For instance, the practice of specifying response length directly addresses the strong correlation identified in prior research [13,20] between output token length and energy consumption. Similarly, avoiding local lingua and terminological ambiguities contributes to clearer semantic meaning in prompts, which Adamska et al. [20] highlighted as being more significant for energy efficiency than simply minimizing prompt length. Variational prompting, by potentially reducing the need for multiple inference calls to explore different solutions, can help minimize the cumulative energy cost, given the per-inference costs detailed in Poddar et al. [13].

4.1. Threats to validity

A range of factors can influence the reliability of our conclusions:

Sample size and diversity. While the results are based on a representative sample of generative AI systems, continued development in this space may limit the generalizability of these

Table 9
Energy savings across llms across Best Practices (BPs).

LLM	Average energy without applying BP (Wh per scenario)	Average energy after applying BP (Wh per scenario)	Average GHG emission without applying BP (g CO_2eq per scenario)	Average GHG emission after applying BP (g CO_2eq per scenario)	Mean of energy and GHG emission reduction
GPT-4o	87.1	52.8	60.9	36.8	39%
Coheres/ Command R+	19.6	10.2	13.7	7.2	48%
Mistral-7B	9.9	5.4	6.8	3.8	45%

Abbreviations: LLM, large language model; GHG, greenhouse gas; g CO_2eq, gram CO_2 equivalent; BP, best practice.

findings to emerging models.

Context-specific results. The results presented may be influenced by the specific use-case contexts and problem scenarios (e.g., coding, writing, travel industry, and finance) under which the experiments were conducted. The findings revealed a substantial decrease in emissions for most test scenarios, but also an increase in a small number of cases, suggesting that variations in conditions can lead to different outcomes. Thus, it is essential to expand the experiments across diverse contexts to validate the generalizability of the results.

Measurement accuracy. The accuracy of GHG emissions measurements is crucial for the validity of the analysis. For proprietary LLMs, such as Azure OpenAI's ChatGPT models, in the absence of officially published data, energy consumption and GHG emissions estimates from Ecologits were approximate, assuming that these models run on an electricity mix in the United States with emissions different from those in the rest of the world. As more accurate and detailed official estimates become available [46], they would help improve the accuracy of estimates.

External factors. External factors that prompt engineering practices, such as changes in hardware efficiency, software updates, and variations in energy sources, could influence results, particularly when LLMs are hosted on unspecified third-party infrastructure. These factors make it challenging to attribute observed differences in energy and carbon reductions across models solely to the implementation of green prompt engineering BPs. Future studies should control for these external variables to isolate the impact of prompt engineering practices across models uniformly.

5. Design framework for generative AI with optimized prompting

A design framework for reducing energy consumption during LLM inference should integrate the prompt engineering practices discussed earlier with key findings from related research [13,15,20]. Reducing energy consumption during LLM inference requires a multifaceted approach that strategically combines optimized prompt engineering with informed model selection and the application of various optimization techniques at both the model and system levels. The prompt engineering techniques of variational prompting, avoiding local lingua and terminological ambiguities, user modeling, specifying response length, and specifying response format all offer significant potential for reducing energy usage, as supported by the experimental findings. These techniques directly address the key factors influencing energy consumption, such as the semantic clarity of the prompt and the length of the generated response, which were highlighted as critical in the research.

In practice, an effective design framework must therefore integrate these insights, recognizing that the most substantial energy reductions will result from a synergistic combination of optimized prompting strategies, careful selection of appropriately sized and architectured LLMs, and the implementation of suitable system-level optimizations. Continuous monitoring and evaluation of energy consumption are crucial for ensuring the long-term sustainability and scalability of LLM deployments, contributing to both environmental responsibility and the economic viability of AI applications.

Fig. 2 presents a high-level design flow of building generative AI applications with optimized prompting. It starts with establishing baseline metrics, optimizing prompt designs, and selecting efficient model configurations. It then integrates prompt-specific optimizations (variational prompting, clear and unambiguous language, role labeling, and strict output specifications) with system-level improvements (quantization, batching, and adaptive controllers) to reduce energy consumption during LLM inference. Optimization techniques, including parallelizing similar queries and reusing frequent outputs, and green prompting approaches, such as speculative decoding and context pruning, together with continuous monitoring of energy and carbon metrics, guide improvements and fine-tuning for sustainable AI operations. By measuring impact at every stage, one can iteratively refine the system and harness additional efficiency gains.

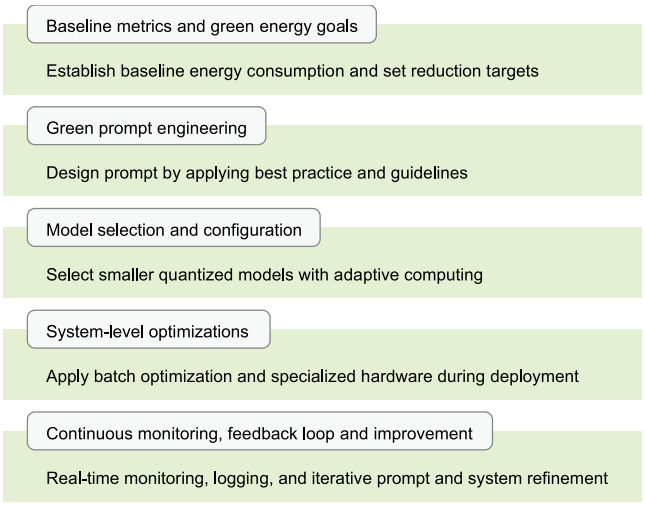

Fig. 2. High-level design of generative AI with optimized prompting.

6. Scalability to real-world enterprise deployment

The best practices are effective across a range of LLM architectures and sizes, allowing for their adoption regardless of the AI platform, model, or vendor. All practices involve changes only at the prompt engineering layer, not at the infrastructure or model level. This means that the BPs can be integrated into existing workflows without major system redesigns or hardware investments. These practices can be embedded into prompt engineering guidelines, templates, or even automated prompt-generation tools, making them suitable for large-scale, repeatable deployment across teams and business units. This framework complements other enterprise-scale efficiency measures (e.g., algorithmic optimizations, such as batching and quantization; carbon-aware scheduling; AI accelerators, such as GPUs and tensor processing units (TPUs); custom application-specific integrated circuits (ASICs); energy-efficient data centers with low power usage effectiveness; optimal usage of storage and compute through resource pooling and elastic scaling; and so on), enabling cumulative gains as organizations scale up their generative AI usage. The framework emphasizes ongoing measurement and iterative refinement, which aligns well with enterprise needs for governance, compliance, and continuous improvement at scale.

However, enterprises often have a wide variety of tasks, domains, and user groups. While the practices are broadly applicable, their effectiveness may vary across contexts, necessitating customization or domain-specific tuning. Scaling these practices requires training prompt engineers, developers, and end users. Enterprises may need to invest in education, documentation, and internal best practice sharing to ensure consistent adoption. Manual prompt optimization can become a bottleneck at scale. To maximize impact, organizations should consider developing or

adopting tools that automate prompt evaluation, suggest optimizations, and monitor energy metrics. Accurate, real-time tracking of energy and emissions at scale may require integration with enterprise monitoring systems or third-party tools, especially when using proprietary LLM APIs.

The study demonstrates that, when applied systematically, these practices can yield 32–52% reductions in energy and carbon emissions per inference—even at scale. This translates to substantial cost savings and progress toward enterprise sustainability goals as generative AI adoption grows. The framework's modular, model-agnostic nature makes it suitable for phased rollouts, pilot projects, and organization-wide adoption.

6.1. Computational costs

Prompt iterations drive cost. Suboptimal prompts often require multiple iterations to achieve the desired output, each triggering a new inference pass. This iterative process significantly increases the computational workload, energy consumption, and associated GHG emissions.

Model size matters. The computational cost is directly influenced by the size of the language model. Larger models (e.g., GPT-4o with ~200B parameters) consume more energy per inference compared to smaller models (e.g., Mistral-7B), due to higher computational complexity and memory requirements.

Empirical measurement. The study uses the Ecologits Calculator to estimate energy consumption and GHG emissions for each prompt scenario across different LLMs. This approach provides a quantifiable link between prompt design choices and computational costs.

Prompt characteristics impact cost. Factors such as prompt length, complexity, and ambiguity can increase the number of tokens processed and generated, further escalating computational demands.

System-level factors. The underlying hardware (e.g., GPU type), batch size, and model architecture also affect computational costs; however, the study isolated the impact of prompt engineering by controlling for these variables in its experiments.

6.2. Advantages of the proposed framework

Significant energy and carbon savings. The empirical results show that integrating the framework's best practices (e.g., variational prompting, avoiding ambiguous terms, specifying response length/format, and role modeling) reduces energy consumption and GHG emissions by 39–48% across multiple LLMs.

Model-agnostic approach. The framework's prompt engineering strategies are effective across different LLM architectures and sizes, making them broadly applicable.

Improved output quality and efficiency. By promoting semantic clarity, concise instructions, and explicit formatting, the framework reduces the need for multiple prompt iterations, resulting in faster, more accurate responses and lower computational overhead.

Synergy with system-level optimizations. The framework is designed to complement hardware and system-level improvements (e.g., batching, quantization, carbon-aware scheduling), thus maximizing overall sustainability gains.

6.3. Limitations of the proposed framework

Not universally optimal. In a small subset of scenarios, certain best practices (e.g., variational prompting) can inadvertently increase energy consumption due to redundant outputs.

Requires manual effort and expertise. Effective prompt engineering still relies on human expertise and trial-and-error, which can be labor-intensive and may limit adoption without further automation.

Context-specific effectiveness. The framework's benefits may vary depending on the task, domain, and LLM used. Some practices are more effective in specific contexts, and their impact can fluctuate with changes in the model architecture or deployment environment.

Measurement and transparency challenges. Accurate, real-time measurement of energy and emissions is challenging, particularly with proprietary LLM APIs, which can impede benchmarking and optimization.

Potential trade-offs with output quality. Aggressively minimizing prompt length or complexity to save energy may sometimes risk omitting necessary context, potentially affecting response quality and requiring follow-up prompts.

Next, these limitations are discussed in light of open research questions and potential directions for future research.

7. Future research challenges and directions in energy-efficient prompting

Generative AI inference is a major driver of AI's growing energy footprint, and the prompt engineering best practices introduced in this study demonstrated positive gains (on average, 39–48% lower CO_2 emissions per inference across models). However, key outstanding issues include making such techniques broadly reliable without degrading model performance, developing tools and metrics to support energy-aware prompt engineering, addressing emerging trade-offs (e.g., between prompt complexity and energy use), and aligning prompt optimizations with evolving model architectures and carbon-aware deployment strategies.

7.1. Open questions and limitations

Generality and robustness*: How universally effective are prompt-efficiency techniques across different tasks and models?* A notable limitation of some BPs is that they are not one-size-fits-all. In testing, a small fraction of scenarios showed increased energy when certain prompting guidelines were applied. For example, variational prompting saved energy in most exploratory tasks, but in ~9% of the cases, it added redundant outputs and, in turn, consumed more energy (a ~50% increase) because the query had only one optimal answer. This underscores that current methods require judicious use; for example, if a scenario does not require analyzing multiple options, forcing variations or additional steps should be avoided. Future research should develop criteria or models to predict when a given prompt strategy will help. This could involve training meta-models to analyze a prompt and determine whether, for example, generating multiple solutions or adding a role persona is likely to yield additional information or merely produce noise. In general, researchers need to validate prompt optimizations across more domains and LLMs to ensure that the observed 40% savings hold broadly. Preliminary work by Rubei et al. [15] showed energy cuts with custom prompt tags in code generation; however, they cautioned that further experiments on different tasks and models are required to confirm these findings. Ensuring that energy-focused prompt tweaks do not degrade output quality or accuracy is an open question. Current studies report no significant performance loss, but a more rigorous evaluation (especially on sensitive tasks) is necessary. Going forward, the field would benefit from a “green prompting” benchmark suite that tests a range of prompt engineering methods across multiple tasks, measuring both energy and task success to identify where techniques generalize or fail.

Automation and tool support*: Can we automate the design and debugging of energy-efficient prompts?* Today, prompt engineering remains a manual, trial-and-error process. This is labor-intensive and requires expertise, which may limit the wider adoption of green prompting practices. A future challenge is to develop automated tools that assist AI developers in crafting low-energy prompts. For example, an intelligent prompt editor could warn if the draft prompt contains ambiguous jargon or unnecessary verbosity that might inflate energy use and suggest a revision. The need for energy-aware debugging frameworks is an important direction for future work. Additionally, a "prompt efficiency analyzer" that can take an arbitrary prompt, run a lightweight inference simulation, or use analytical models (perhaps leveraging the linear correlation between response latency and energy), and then output guidance: e.g., "This prompt might cause extra iterations because the term 'XYZ' may be unclear to the model—consider clarifying or using a standard term." Another helpful tool would be an automated prompt optimizer that can rewrite a given query in multiple ways and test which version yields the desired answer with minimal tokens or computation. Recent research directions support this idea. For instance, Soham Poddar et al. [13] found that inference time correlates strongly with energy use, suggesting that inference time can serve as a proxy metric for quickly evaluating prompt efficiency. Combining such insights with large-scale language feedback, an AI system can tunes prompts for minimal energy while preserving their intent, by iteratively shortening or simplifying the prompt and checking that the model's answer remains correct. Developing such design frameworks and Integrated Development Environment plugins for sustainable prompt engineering is a promising area for future work. This will require collaboration between NLP experts and software tooling researchers to integrate energy metrics into the prompt development cycle.

Evaluation metrics and transparency. Underpinning the above is the challenge of accurately and consistently measuring energy consumption. Today, researchers often rely on indirect estimates (e.g., the EcoLogits tool and the assumed carbon intensity of power grids) or proxy metrics, such as latency, because direct energy data from proprietary LLM APIs are rarely available. This lack of transparency makes it harder to benchmark improvements or set concrete targets. An open question is how to establish standard evaluation protocols for "green prompts." Future work could push for energy usage reporting features in LLM APIs (so that models can return an estimated joule cost along with each answer) or develop more advanced estimation techniques. One recent approach, LLMCO2 [27], aims to accurately predict the carbon footprints of LLM inference using performance counters. Reliable energy metrics would enable multi-objective optimization; that is, we could require that a prompt variant not only meet an accuracy threshold but also remain below a specified energy budget. Additionally, research should examine the trade-off between efficiency and other ethical metrics. For example, does aggressively minimizing prompt length risk omitting context that avoids biases or errors? If a concise prompt saves energy but occasionally causes a factual mistake that needs correction in a follow-up prompt, the net gain could be lost. Developing multifaceted evaluations (combining energy, accuracy, and fairness) is thus an emerging consideration for sustainable AI.

7.2. Emerging concerns and new research directions

Domain-specific models and knowledge*: Can leveraging domain expertise reduce the energy cost of prompting?* Current prompting practice typically relies on a large general model for all queries, even when many contain domain-specific terminology or require specialized knowledge. This can be inefficient because a large model may require more computation to infer the domain context, whereas a smaller, domain-specific expert model could determine it with fewer computations. A future direction is to incorporate greater domain awareness into the prompting process to reduce unnecessary token usage and iterations. One approach is to incorporate domain-specific LLMs or modules [47–51]. For instance, instead of always relying on a general LLM, a query about a medical case could be routed to a compact medical domain model (such as BioGPT [47]) or augmented with key domain facts retrieved from a knowledge base. By using hybrid architectures (general LLM + domain experts), the system could avoid over-explaining basic domain terms in the prompt or output, thus saving tokens. Research is needed to determine the best ways to integrate such domain expertise. One idea is in-context knowledge injection, whereby a prompt is automatically expanded with relevant facts or definitions (saving the model from lengthy reasoning). Another approach is a cascade of models: for example, a lightweight financial model could answer simple finance questions at low cost, escalating to a larger model only when the query is complex. This relates to the broader vision of energy-proportional AI, in which computational effort is matched to query difficulty. However, realizing this in practice will require research into model interoperability and response fusion (to maintain answer quality when multiple models are involved).

Prompt complexity vs. efficiency trade-offs*: How can we balance the need for complex reasoning in prompts with the goal of minimizing energy?* An emerging concern is that some of the most powerful prompting techniques entail additional verbosity or computational demands. For example, chain-of-thought prompting (in which the model is asked to "think step by step") often improves accuracy on reasoning tasks, but it produces much longer outputs and uses more tokens, potentially increasing inference energy consumption significantly. A recent study by Riccardo Rubei et al. [15] compared zero-shot, one-shot, few-shot, and chain-of-thought prompting for code generation tasks and found that different prompt strategies led to widely varying energy costs. Multistep reasoning prompts may require the model to perform internal computations that a direct prompt would skip. This raises a key research question: Can we obtain the benefits of techniques such as chain-of-thought without full energy overhead? Possible directions include training LLMs to perform reasoning internally (so that the prompt does not have to explicitly enumerate every step) or using smaller "reasoning helper" models that run alongside the main LLM to offload some work. Another idea is to use partial-completion prompts: instead of having the LLM explain every step in natural language (which consumes tokens), it could output a succinct reasoning trace in a coded form that is more token-efficient, which is then parsed to yield the final answer. Moreover, for simpler queries, a minimal prompt could be used, with automatic switching to a chain-of-thought style for complex prompts. Research on measuring task complexity and dynamically adapting prompt strategies could be helpful. Optimizing energy must be balanced against accuracy; therefore, future prompt frameworks may need to support adaptive verbosity, expanding the prompt only as much as needed to solve the problem at hand.

Evolution of LLM architectures*: How will new model architectures and sizes influence prompt efficiency strategies?* The landscape of LLMs is rapidly changing; models are getting larger in some cases but also more efficient and specialized in others. Transformer-based LLMs currently dominate, but alternative architectures are emerging that promise different performance–efficiency trade-offs. For example, the RWKV [32] model and others inspired by recurrent networks aim to retain long context with a lower compute cost, and

Megalodon [33] proposes transformer variants for unlimited context lengths. If these architectures take hold, some prompt guidelines may need to be revisited. A research question is how prompt length or formatting, for example, affects a streaming recurrent model versus a standard transformer. Conversely, models with extremely large context windows (e.g., 100k tokens) might encourage users to add more information to prompts (to avoid multiple queries), potentially increasing inference work. Thus, prompt efficiency research must co-evolve with model development. One important direction is to test the existing BPs on a variety of model types—from small 7B-parameter models to multi-hundred-billion-parameter models, including new architectures—to assess whether the energy savings hold. Early evidence suggests that model size matters; for example, smaller models, such as Mistral-7B, sometimes achieved higher relative energy savings (50%+) than larger models. We need to understand such phenomena more deeply. Additionally, future LLMs might themselves become more energy aware. Researchers have called for "energy-adaptive LLMs" [20] that can adjust their computation based on the prompt—for instance, skipping some layers or using lower precision if a prompt appears easy to answer, or dynamically limiting the response length to save energy.

Deployment and infrastructure strategies*: How can we deploy LLMs in a more carbon-efficient way, and what role does prompt engineering play in that?* Beyond the prompts and models, where and when inference runs influence the carbon footprint. Recent research by Chien et al. [19] demonstrated that intelligently routing user requests to data centers with cleaner energy can cut inference emissions by ~35% today (and up to ~56% by 2035, as grids get greener). This strategy, CarbonMin, is effectively a form of "carbon-aware load balancing." One challenge is integrating such strategies with prompt-level optimizations. For example, if a prompt is likely to trigger a highly energy-intensive response (e.g., a complex computation), a system could automatically forward that query to run in a region where renewable energy is currently abundant (or at a time when the grid is cleaner), thereby minimizing its carbon footprint. Conversely, simple prompts might be handled locally or on edge devices to avoid network overhead. Future LLM systems might dynamically decide not just *how* to prompt a model but where to run the model. This raises research questions around scheduling and latency. Another deployment-level approach is to use multi-model cascades for efficiency, which involve deploying a hierarchy of models and using only the large, energy-intensive model when necessary. This has been explored in prior work on model serving and can be informed by prompt engineering [52]; for example, a lightweight model first assesses whether its confidence (i.e., the quality of its answer relative to the prompt requirements) is insufficient, and a more powerful model is then engaged. Designing orchestration algorithms that balance energy, carbon intensity, and response quality is a complex systems problem for future research. Additionally, advances in hardware will influence deployment. New AI accelerators [21] promise higher inference throughput per watt, and fully leveraging them might require prompt formats that align with hardware-friendly processing (for instance, certain formats might enable better use of batching or caching on accelerators). Researchers should therefore explore the interaction between prompt structures and hardware utilization; for example, can prompts be optimized to make transformer inference more cache-efficient?

Agentic communication. Emerging agentic AI systems [53] could benefit significantly from adopting the proposed BPs for drafting prompts. These systems, which rely on precise and efficient communication with LLMs, can leverage well-crafted prompts to elicit optimal responses swiftly and accurately. By minimizing the need for iterative cycles of prompt refinement and response correction, these BPs not only enhance the quality of the generated outputs but also yield substantial energy savings. The streamlined prompt–response process reduces computational overhead, thereby making agentic AI systems more energy-efficient.

In summary, addressing these challenges will require an interdisciplinary effort: advances in NLP (for smarter prompts and models), software engineering (for tooling and integration), and systems design (for deployment and measurement). By tackling open issues, from automating prompt optimizations to exploiting new model capabilities and carbon-aware infrastructure, we move closer to the goal of sustainable, scalable generative AI—systems that deliver rich AI capabilities while minimizing their environmental impact. The ongoing convergence of ideas, such as green prompting, carbon-aware computing, and adaptive inference, promises an exciting research frontier in which prompt design becomes a key lever for both AI performance and sustainability [20].

8. Conclusion

This paper examined the complexities of prompt engineering in generative AI systems, emphasizing the importance of energy-efficient practices to enhance sustainability. Suboptimal prompts can lead to multiple iterations, thereby increasing energy consumption. Energy-efficient prompt engineering practices (e.g., concise prompts with high semantic precision) can help reduce the environmental impact of LLM inferences.

The paper presented several best practices to enhance energy efficiency, including variational prompting, avoiding local lingua, avoiding terminological ambiguities, user modeling, specifying optimal response length, and providing the desired response format.

Experiments were conducted to validate the effectiveness of these best practices, showing energy savings and reductions in GHG emissions across multiple LLMs. Each best practice demonstrated varying degrees of energy savings, with overall reductions in GHG emissions ranging from 39% to 48%. However, some practices led to increased emissions in a small subset of test scenarios.

CRediT authorship contribution statement

Sanjay Podder: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Project administration, Methodology, Investigation, Formal analysis, Data curation. **Hema Date:** Validation, Supervision, Conceptualization. **Shankar Murthy:** Validation, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This research was conducted as part of the doctoral research program at the Indian Institute of Management Mumbai. The study did not receive any specific external grant from funding agencies in the public, commercial, or not-for-profit sectors.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.ese.2026.100684.

References

[1] What is Generative AI and How does it Impact Businesses?" BCG, Link.
[2] Fiona Fui-Hoon Nah, et al., Generative AI and ChatGPT: applications, challenges, and AI-human collaboration, Journal of information technology case and application research 25.3 (2023) 277–304. Link.
[3] Narendra Babu Mundlamuri, A Comprehensive Guide to Prompt Engineering, DZone, 2024. Link.
[4] What Is Prompt Engineering?" IBM, Link.
[5] Google Cloud. "Prompt Engineering: Overview and Guide." Link.
[6] Banghao Chen, et al., Unleashing the Potential of Prompt Engineering in Large Language Models: a Comprehensive Review, 2025 arXiv,Link.
[7] Hai Dang, et al., How to prompt? Opportunities and challenges of Zero- and few-shot learning for Human-AI interaction in creative applications of generative models, arXiv (2022). Link.
[8] J.D. Zamfirescu-Pereira, et al., Why johnny can't prompt: how non-AI experts try (and fail) to design LLM prompts. Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems, Link, 2023.
[9] Qianou Ma, et al., What should we Engineer in Prompts? Training Humans in Requirement-Driven LLM Use, *arXiv:2409.08775*, 2024. Link.
[10] John Johnson, Small Language Models (SLM): a Comprehensive Overview, 22 Feb 2025. Link.
[11] Golam Md Muktadir, A Brief History of Prompt: Leveraging Language Models. (Through Advanced Prompting), 2023 arXiv, Link.
[12] Aldeida Aleti, Software testing of generative AI systems: challenges and opportunities. 2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE), IEEE, 2023. Link.
[13] Soham Poddar, et al., Towards Sustainable NLP: Insights from Benchmarking Inference Energy in Large Language Models, arXiv:2502.05610, 2025. Link.
[14] Erik Johannes Husom, et al., The Price of Prompting: Profiling Energy Use in Large Language Models Inference, *arXiv:2407.16893*, 2024. Link.
[15] Riccardo Rubei, et al., Prompt Engineering and its Implications on the Energy Consumption of Large Language Models, arXiv:2501.05899, 2025. Link.
[16] F. Caravaca, et al., From Prompts to Power: Measuring the Energy Footprint of LLM Inference, arXiv:2511.05597, 2025. Link.
[17] H. Kim, J. Ben-Othman, Eco-friendly low resource security surveillance framework toward green AI digital twin, IEEE Commun. Lett. 27 (1) (Jan. 2023) 377–380, https://doi.org/10.1109/LCOMM.2022.3218050. Link.
[18] C. Elsworth, et al., Measuring the Environmental Impact of Delivering AI at Google Scale, arXiv:2508.15734, Aug. 2025. Link.
[19] Andrew A. Chien, et al., Reducing the carbon impact of generative AI inference (today and in 2035). Proceedings of the 2nd Workshop on Sustainable Computer Systems, Link, 2023.
[20] Marta Adamska, et al., Green Prompting, arXiv:2503.*10666*, 2025. Link.
[21] Christoforos Kachris, A survey on hardware accelerators for large language models, Appl. Sci. 15 (2) (2025) 586. Link.
[22] Heming Xia, et al., Unlocking efficiency in large language model inference: a comprehensive survey of speculative decoding. Findings of the Association for Computational Linguistics, ACL, 2024, pp. 7655–7671. Link.
[23] James O'Donnell, Casey Crownhart, We did the math on AI's energy footprint. Here's the story you haven't heard, MIT Technology Review (May 2025). Link.
[24] Andrew A. Chien, GenAI: giga$$$, TeraWatt-Hours, and GigaTons of CO2, Commun. ACM 66 (8) (2023), 5-5. Link.
[25] Siddharth Samsi, et al., From words to watts: benchmarking the energy costs of large language model inference. 2023 IEEE High Performance Extreme Computing Conference (HPEC), IEEE, 2023. Link.
[26] Pratyush Patel, et al., Splitwise: efficient generative LLM inference using phase splitting. 2024 ACM/IEEE 51st Annual International Symposium on Computer Architecture (ISCA), IEEE, 2024. Link.
[27] Zhenxiao Fu, et al., LLMCO2: Advancing Accurate Carbon Footprint Prediction for LLM Inferences, arXiv:2410.02950, 2024. Link.
[28] A.I. Smartly, What is the CO2 Emission per Chatgpt Query?, Jun 7, 2024. Link.
[29] Sasha Luccioni, Yacine Jernite, Emma Strubell, Power hungry processing: watts driving the cost of AI deployment?. The 2024 ACM Conference on Fairness, Accountability, and Transparency, 2024. Link.
[30] Jinhao Li, et al., Large Language Model Inference Acceleration: a Comprehensive Hardware Perspective, *arXiv:2410.04466*, 2025. Link.
[31] Haoyang Li, et al., A Survey on Large Language Model Acceleration Based on KV Cache Management, *arXiv:2412.19442*, 2025. Link.
[32] PCF-RWKV: Large Language Model for Product Carbon Footprint Estimation. *Link*.
[33] Megalodon: Efficient LLM Pretraining and Inference with Unlimited Context Length. Link.
[34] Radosvet Desislavov, Fernando Martínez-Plumed, José Hernández-Orallo, Trends in AI inference energy consumption: beyond the performance-vs-parameter laws of deep learning, Sustainable Computing: Informatics and Systems 38 (2023) 100857. Link.
[35] H.-Y. Lin, J. Voas, Lower energy large Language models (LLMs), Computer 56 (10) (Oct. 2023) 14–16, https://doi.org/10.1109/MC.2023.3278160. Link.
[36] Aaron Hurst, et al., GPT-4o System Card, *arXiv:2410.21276*, 2024. Link.
[37] Grant Wilkins, Keshav Srinivasan, Richard Mortier, Offline energy-optimal LLM serving: workload-Based energy models for LLM inference on heterogeneous systems, ACM SIGENERGY Energy Informatics Review 4.5 (2024) 113–119. Link.
[38] OpenAI. Prompt engineering: Enhance Results with Prompt Engineering Strategies, Link.
[39] Simon Arvidsson, Johan Axell, Prompt Engineering Guidelines for Llms in Requirements Engineering, 2025. Link.
[40] OpenAI. Best Practices for Prompt Engineering with the Openai API. Link.
[41] Roberto Verdecchia, et al., A systematic review of green AI, Wiley Interdiscip. Rev., Data Min. Knowl. Discov. 13 (4) (2023) e1507. Link.
[42] LLaMA, Llama 3.1 Model Cards and Prompt Formats, 2024. Link.
[43] Albert Q. Jiang, et al., Mistral 7B, *arXiv:2310.06825*, 2023. Link.
[44] Ecologits-Calculator. Link Accessed 10-January-2025.
[45] ISO 14044, Environmental Management - Life Cycle Assessment - Requirements and Guidelines, 2006. Link.
[46] Sam Altman. Blog - the Gentle Singularity. Accessed on June 20, 2025. Link.
[47] Y. Luo, Y. Sun, Y. Wang, et al., Biogpt: Generative Pre-trained Transformer for Biomedical Text Generation and Mining, Microsoft Research, 2022. Link.
[48] Hongyang Yang, Xiao-Yang Liu, Christina Dan Wang, Fingpt: Open-Source Financial Large Language Models, arXiv preprint, arXiv:2306.06031, 2025. Link.
[49] K. Singhal, A. Azizi, T. Tu, et al., Large language models encode clinical knowledge, Nature 620 (2023) 172–180. Link.
[50] Ilias Chalkidis, et al., LEGAL-BERT: the muppets straight out of law school. Findings of the Association for Computational Linguistics: EMNLP 2020, 2020. Link.
[51] Asma Ben Abacha, et al., MEDEC: a Benchmark for Medical Error Detection and Correction in Clinical Notes, *arXiv:2412.19260*, 2025. Link.
[52] Lingjiao Chen, Matei Zaharia, James Zou, Frugalgpt: How to Use Large Language Models While Reducing Cost and Improving Performance, *arXiv:2305.05176*, 2023. Link.
[53] Yuheng Cheng, et al., Exploring Large Language Model Based Intelligent Agents: Definitions, Methods, and Prospects, arXiv:2401.03428, 2024. Link.